The All-Hex Geode-Template for Conforming a Diced Tetrahedral Mesh to any Diced Hexahedral Mesh

Scott A. Mitchell[1]

Abstract. *Take a hexahedral mesh and an adjoining tetrahedral mesh that splits each boundary quadrilateral into two triangles. Separate the meshes with a buffer layer of hexes. Dice the original hexes into eight, and the tetrahedra into four hexahedra. Then I show that the buffer layer hexes can be filled with the* geode-template*, creating a conforming all-hex mesh of the entire model. The geode-template is composed of 26 hexahedra. The hexahedra have acceptable quality, depending on the geometry of the buffer layer. The method used to generate the geode-template is general, based on interleaving completed dual surfaces, and might be extended to other transition problems.*

Figure 1. The all-hex geode-template: Left, the diced tet and hex interface; middle, the heart of the geode; right, the boundary of the template.

keywords. mesh generation, hexahedra, tetrahedra, conforming, transition

1. Introduction

For some FEM calculations, hexahedral meshes are preferred to tetrahedral meshes. However, for a geometrically complicated domain, automatically generating a hexahedral mesh is much more difficult than generating a tetrahedral mesh. A hexahedral mesh can always be obtained from a tetrahedral mesh by *dicing*, dividing each tetrahedra into four hexahedra, but the resultant mesh quality is relatively poor.

At Sandia National Laboratories, SNL, we often want to analyze models composed of hundreds of simple parts, arranged together in a complicated way. In many models, there is a potting material surrounding these parts whose geometry is the complement of the union of the other parts. This potting material is often the most difficult to mesh, but the least interesting to the analyst. Also, at SNL and in industry, we commonly have a complicated part that may be mostly decomposed into simple pieces, with one or two difficult nuggets remaining. The mesh of the entire model should be *conforming*, that is, the mesh should be a simplicial complex, with elements meeting face-to-face, edge-to-edge, and node-to-node.

For these scenarios, a reasonable solution is to first (automatically) mesh the simple or important parts with high-quality hexahedra, then automatically mesh the complicated or unimportant parts with tetrahedra. On the surfaces between hex-parts and tet-parts the quads can be divided into two triangles; many Delaunay-based tetrahedral meshers

1 samitch@sandia.gov, http://endo.sandia.gov/~samitch. Parallel Computing Sciences Dept., Sandia National Laboratories, Albuquerque, NM 87185. The author was supported by the Mathematical, Information and Computational Sciences Division of the U.S. Department of Energy, Office of Energy Research, and works at Sandia National Laboratories, operated for the U.S. DOE under contract No. DE-AL04-94AL8500. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the U.S. DOE.

can generate a tet mesh that conforms to these triangles. Hence the tetrahedral and hexahedral meshes will conform node-wise, but each quadrilateral will have a diagonal edge cutting it into two triangles.

The problem is what to do with this non-conforming interface. One solution is *tied contacts*, modifying the analysis software to handle a non-conforming mesh by interpolating between intermediate FEM solutions along the interface. Besides complicating the software, in some situations the running time and analysis errors are unacceptably high. Another solution is to introduce square pyramid elements. The interface is now conforming, but again the analysis code must be modified, this time to handle square pyramid elements; many SNL analysis codes do not even support the more common tetrahedral elements.

In this paper I propose a new solution which generates a conforming mesh for the entire model composed entirely of hexahedra. First, the tetrahedral mesh is geometrically shrunk away from its interface with the hex mesh, creating a boundary layer of hexahedra. (The boundary layer could be created before the tet-mesh. Note that for some interface geometries, e.g. Figure 9 right, shrinking is impossible.) Each hex of the boundary layer has the same structure: the top quadrilateral face is shared with the tet mesh and divided into two triangles, the opposite bottom face is shared with the hex mesh, and the remaining four side faces are shared with other hexes of the boundary layer.

Second, the hexahedral mesh outside the boundary layer is diced by dividing each hex into 8, and the tetrahedral mesh inside the layer is diced into hexes. Boundary layer "hexes" now have six quads on the diced-tet-mesh interface and four quads on the diced-hex-mesh interface. Third, each boundary layer hex is filled in with the *geode-template*, which is composed of 26 hexahedra and conforms to the diced meshes. Each of the four side faces have the same quad mesh, so templates match up and the entire mesh is conforming.

There are several options and applications: The entirety of a solid-model part need not be meshed with tetrahedra. For example, Plastering could fill much of the volume, and tet-meshing fill only the remaining voids; see Meyers[4] and Tuchinsky[11]. There are a number of issues related to tool infrastructure and finding good geometric positions for the boundary layer as well.[10] Finding a hex-template for slightly different situations has been studied for some time. The most famous instance is Schneiders's open problem, which has an interesting history and is accessible from the web.[9] Two template problems with similar structure arise in Eppstein's hex-mesh existence proof.[1]

The method of generating the geode-template is general and could be extended to other transition problems. The method is based on the Spatial Twist Continuum, STC [6][7], the surface arrangement dual to a hex mesh. In particular, it is based on a new form of Whisker Weaving's *sheet* (surface) moving[2]: First I create a completion of the STC of the diced tet mesh, then a separate completion of the STC of the diced hex mesh. I then push these two arrangements together so that they intersect, combining the STCs. Dualize the combined STC creates the hexes of the geode-template.

The advantage of the geode-template is that all-hex meshing is automated and analysis codes do not need to be modified. One drawback is that the element count is high because of the dicing step. Also, the quality of the worst-element is probably worse than in a diced-tet-mesh of the entire model. This is offset by the geode-template allowing excellent-quality structured meshes in areas of interest.

The remainder of this paper is organized as follows. Section 2 describes the general transition method used to create geode-templates. Section 3 describes the hexes of the geode-template and comments on mesh quality. Section 4 presents some preliminary examples of models meshed with the geode algorithm. I discuss in section 5 extending the geode-template construction method, and conjecture why filling certain other hex-templates is difficult. Section 6 presents conclusions. Appendix A summarizes node-position and hex-connectivity for those wishing to reconstruct the geode-template.

2. General Transition Method

In this section I describe a general method for making two adjacent meshes conform by dicing them and inserting a transition layer. First, in section 2.1, I illustrate the process when the two adjacent meshes arise from a triangular mesh adjacent to a quadrilateral mesh, the two-dimensional geode-template. In section 2.2 I describe the construction of the three-dimensional geode-template.

2.1 Two-dimensional Geode-Template Construction

The two dimensional version of the conforming transition problem is trivial: The 1-dimensional elements, edges, are the same in triangular and quadrilateral meshes, so they already conform. In three-dimensions, the interface is non-conforming: Each interface-quad is subdivided into two triangles. Carrying out the geode-formation process in two-dimensions aids in understanding the three-dimensional case.

First, I separate the interface between the triangular and quad mesh. This means topologically splitting the shared nodes and edges into two, and geometrically shrinking the triangular mesh away from the quad mesh; see Figure 2. (Alternatively, the interface could be separated before the triangular mesh is generated.)

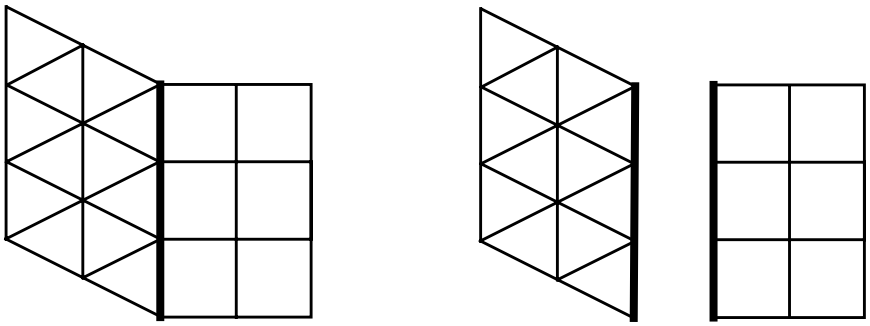

Figure 2. First, the interface is separated. In these pictures the meshes are structured, but both meshes can be unstructured.

Second, I dice the meshes, then complete the dual curves for each diced mesh separately; see Figure 3. Dicing the triangular mesh is necessary in order to convert triangles into quads. More subtly, dicing both meshes ensures that the dual curves can be completed in pairs. I join the two curves that arise from dicing an interface quad. Note that the dual of a diced triangular mesh is composed of circular curves, except where the interface interrupts it. For the triangular mesh I join curves to complete these circles: For a node N of the original triangular mesh on the interface, it has two original interface edges, E_1 and E_2. Each interface edge gets diced in two, creating E_{11}, E_{12}, E_{21}, and E_{22} in order. I join the ends of the curves dual to E_{11} and E_{22}, the diced edges not containing N, as in Figure 3.

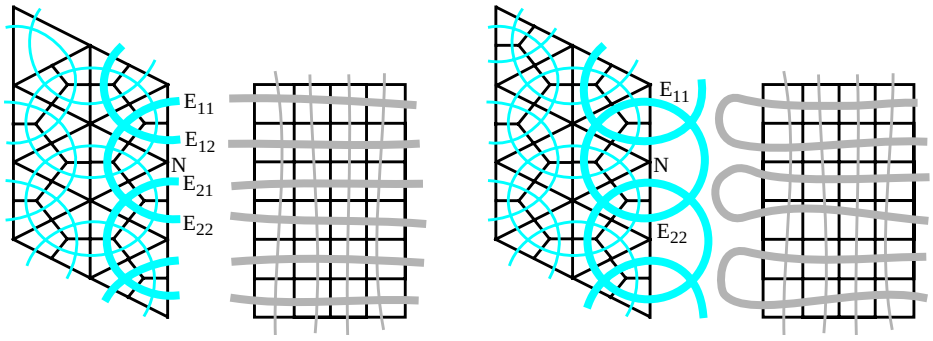

Figure 3. Second, left, the triangular and quadrilateral meshes are diced and the dual curves constructed. Right, the STCs are then completed for each mesh separately.

Third, I push the completed duals together so that they overlap; see Figure 4. Dualizing creates the primal mesh.

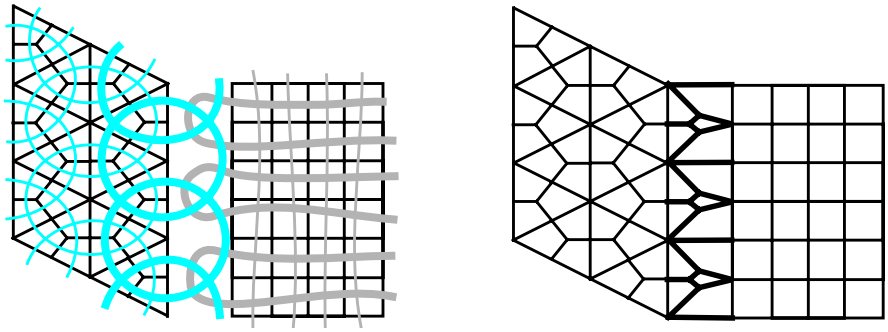

Figure 4. Left, the completed duals are pushed together so that they overlap. Right shows the resulting primal mesh.

2.2 Three-dimensional Geode-Template Construction

The same principles used in constructing the two-dimensional geode-template apply to the three-dimensional case. The main difference is the structure of the completed STCs. Instead of just completing curves, I must complete surfaces. For the hexahedral mesh, its interface mesh is composed of quads. The dual curves of the interface mesh are closed curves. Dicing converts each curve into two identical, parallel copies. I complete the diced hex mesh dual by making a tunnel surface for each parallel pair. Note that at each pre-diced quad, two tunnels pass through each other; see Figure 5. Since the arrangement must be in general position in order to dualize to a hexahedral mesh[5], I must chose which tunnel's roof is higher than the other. The choice is arbitrary; there does not need to be any consistency between neighboring pre-diced quads. But this does introduce an asymmetry within the template.

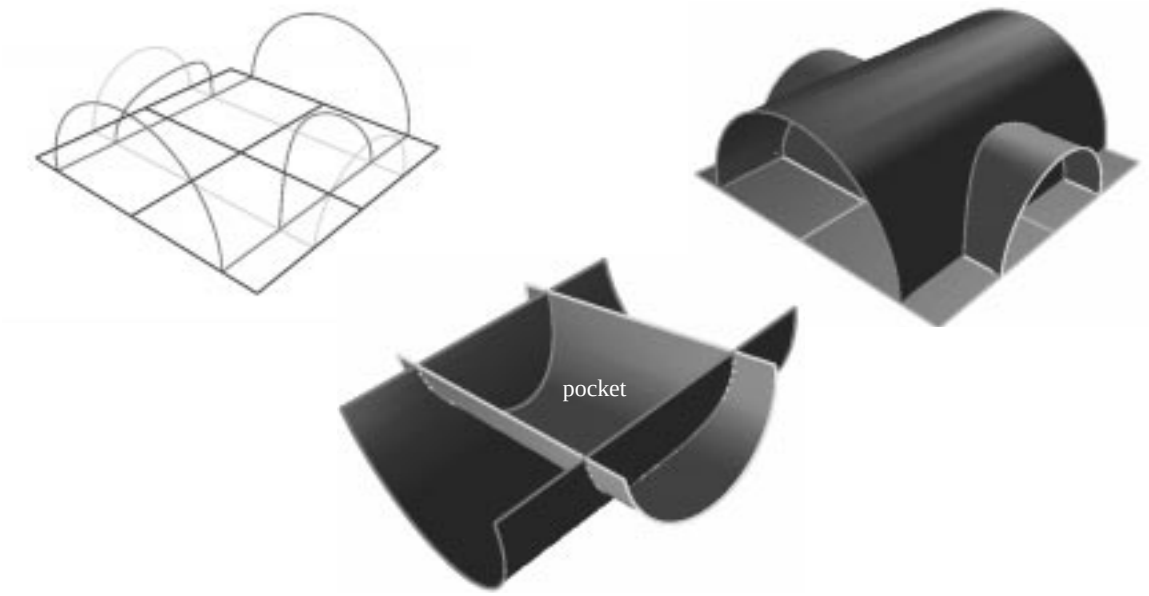

Figure 5. A local view of the completed dual surfaces for the diced hexahedral mesh. Right and left, the bottom four quadrilaterals are the diced hex interface mesh. Center, the quads have been removed and the tunnels turned upside-down. I use cylinders of different radii only to illustrate that one tunnel locally passes below the other; topologically, the tunnels always match up exactly with tunnels of neighboring templates.

For the tetrahedral mesh, its interface mesh is composed of triangles. Dicing produces circle-curves on the interface[6], see Figure 3. I complete these circles into spheres; see Figure 6. Considering a section of the surfaces above a pre-diced

quad, note that two of the completed circles intersect in a curve that starts and ends on two quads of the diced triangular mesh.

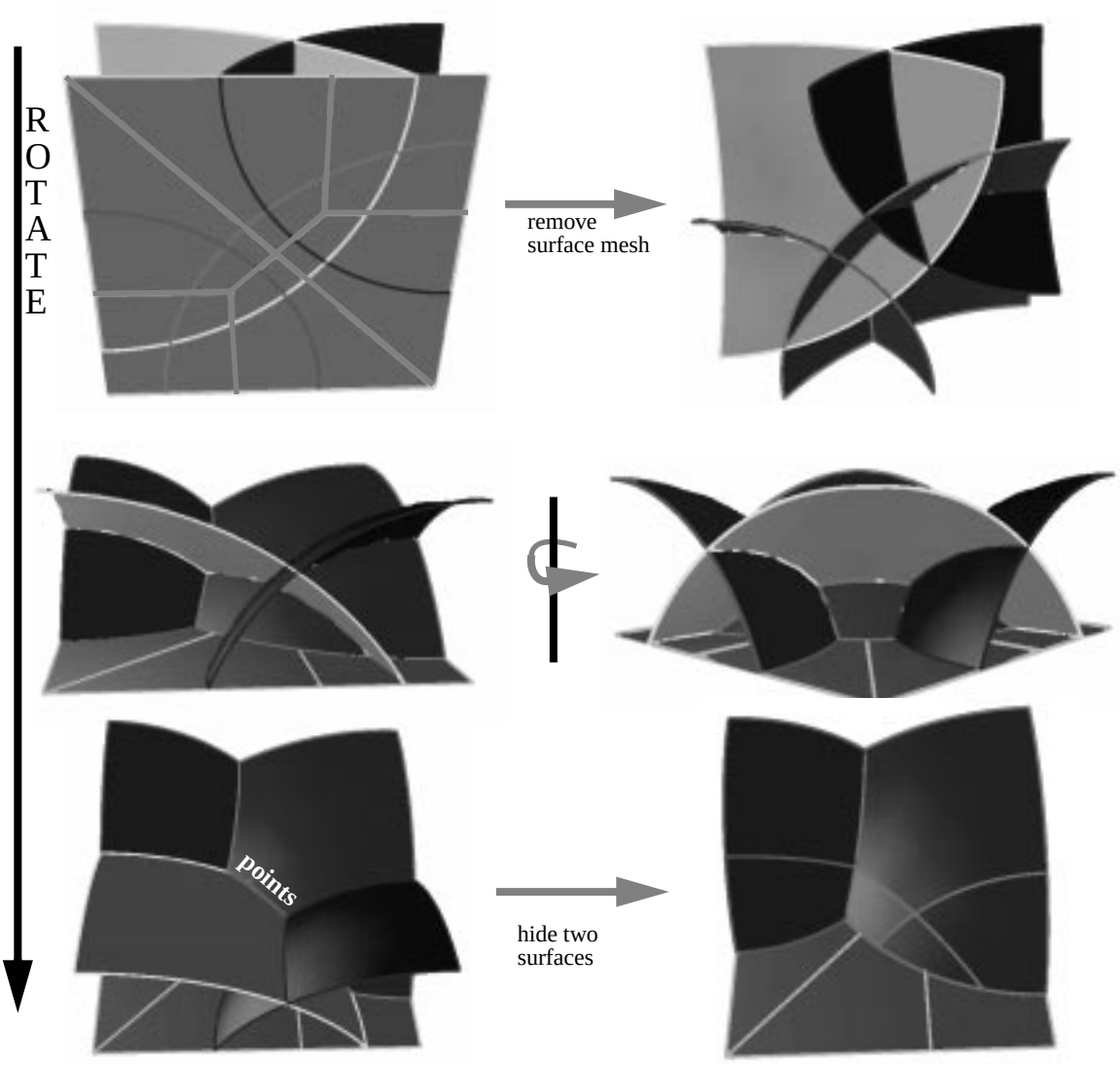

Figure 6. The completed dual surfaces for the diced tetrahedral mesh.

I now push the completed duals together. I can choose how far to overlap the duals. There are two key features of my choice: First, the two points where three diced-tet surfaces intersect, as in Figure 6 bottom left, lie inside both tunnels, the *pocket* in Figure 5 center. Second, each of the curves of intersection between two diced-tet surfaces, except the curve that starts and ends on the interface mesh, exit the side of the template inside one of the arch-like tunnel entrances, so that the side of the three-dimensional template looks like the two-dimensional template as in Figure 4 or Figure 1 center. See Figure 7 for views of the entire arrangement.

I am fortunate that the arrangement dualizes to a well-defined hexahedral mesh, with no degenerate elements, without any fix-ups[5][2] needed. The next section discusses the geode-templates hex structure and quality.

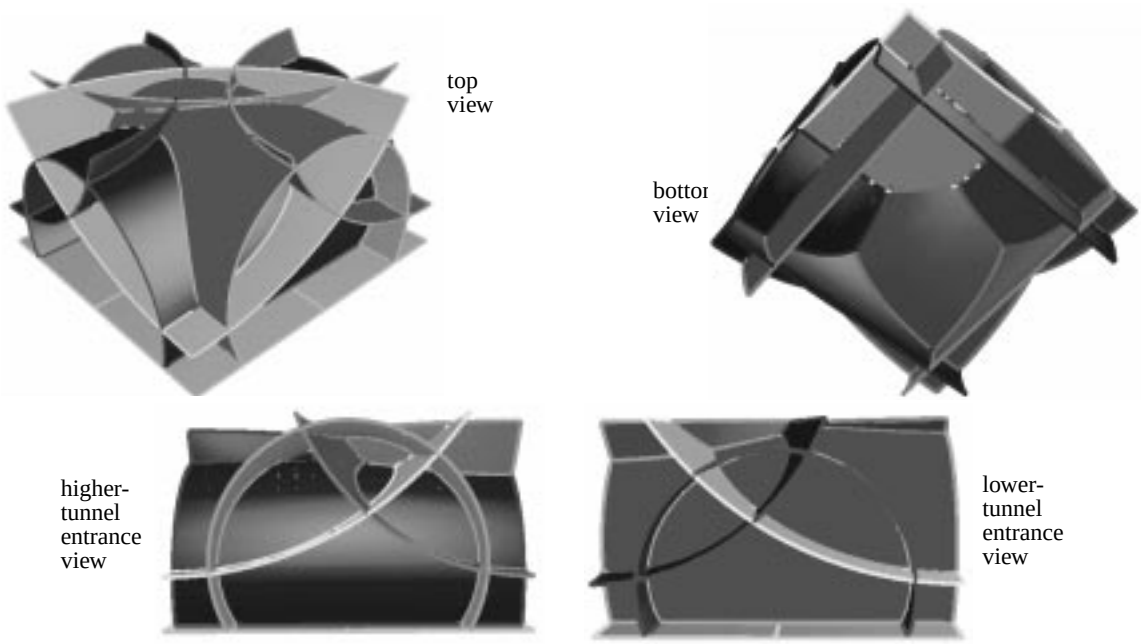

Figure 7. The total arrangement of the pushed-together diced-tet and diced-hex completed duals. Note that looking into the entrance of the higher tunnel is different than looking into the entrance of the lower tunnel; compare the lower left of the figure with the lower right.

3. Geode-Template Hexes

3.1 Hex Structure

Since the mesh is highly-unstructured, the traditional views of layers through z-planes are not very informative. Removing hexes from the outside-in is slightly better, such as Figure 1 center. The most useful static views I have found are hex layers dual to each surface of the STC. Note that the layers from the diced-hex-mesh and the diced-tet-mesh intersect, but the layers are separate. Each tunnel is radially symmetric about a vertical line through the center of the template. To truly understand the hexes of the template, I suggest creating a computer model, or even better a physical toothpick or pipe-cleaner model, based on Appendix A. An interesting feature is that in these models the hexes appear to rotate by 45 degrees when travelling from the diced-triangle top to the diced-quad bottom.

3.2 Mesh Quality

The quality of the geode-template obviously depends on the shape of the buffer layer. The best-quality hexes appear to be achieved with a short geode template. This is serendipitous, since it means that the buffer layer can be thin. Let the *scaled jacobian* at a node of a hex be defined as the triple product of the vectors along the three edges of the node, divided by the product of the lengths of the vectors. The scaled jacobian for a hex is the minimum scaled jacobian among its nodes. For a geode-template inside a rectangular parallelepiped with a square base of length 24 and height 14, and nodes positioned with optimized-jacobian based smoothing[3], the scaled jacobian ranges from 0.26 to 0.53.

A complete quality summary follows in Table 1, “CUBIT quality report for a 24 x 14 x 24 geode-template.” Quality measures are defined in Robinson.[8]

Table 1: CUBIT quality report for a 24 x 14 x 24 geode-template.

Quality Measure	Average	Std. Dev.	Minimum	Maximum
Aspect Ratio	1.75	0.55	1.07	3.07
Skew	0.54	0.16	0.17	0.76
Taper	0.30	0.13	0.038	0.76
Element Volume	300	223	97	786
Stretch	0.41	0.13	0.25	0.63
Diagonal Ratio	0.53	0.09	0.38	0.74
Dimension	3.19	0.94	1.97	5.00
Jacobian	6.99	44	26	164
Scaled Jacobian	0.34	0.09	0.26	0.53

4. Examples

This section gives some preliminary examples of meshes created in CUBIT with the geode-algorithm.[10] This algorithm blends the geode-template with MSC’s ARIES tet mesher and CUBIT’s Plastering[4] hex-dominant mesher. As algorithms for positioning nodes mature I expect mesh-quality to increase.

The first example is from Clay Fulcher; see Figure 8. It consists of a simple geometry, a cube, but with imprinted circles of different sizes on three sides that prevent 2.5-dimensional meshing and make it difficult to decompose. This nugget is surrounded by large, sweepable parts. The geode algorithm successfully produces an all-hex mesh.

In Fulcher’s example I place the transition layer directly on the geometry, without Plastering any of the volume. This produces 6864 geode-template hexes, with worst scaled-jacobian 0.017 and worst aspect ratio 11. There are 6140 diced-tet hexes, with worst scaled-jacobian 0.038 and worst aspect ratio 10.

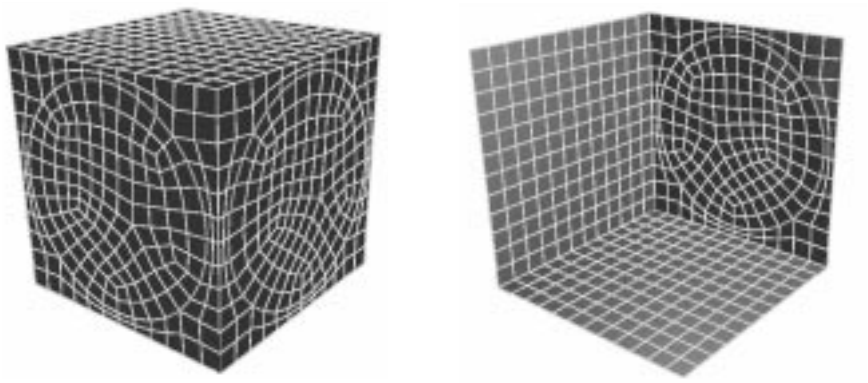

Figure 8. The frontal (left) and cut-away (right) view of Clay Fulcher’s problem after dicing. Despite the simple geometry, a highly-unstructured hex-mesh is necessary.

The next example is the square pyramid of Schneiders's open problem; see Figure 11. There are 512 hexes, with worst scaled-jacobian 0.033 and worst aspect ratio 9.4. I chose this example to be cute; I'm filling the pyramid transition element with the geode transition template. The geode-template solution probably does not have any practical value and does not solve the open problem because dicing is required.

Another type of example that would demonstrate the geode-template's utility is the cube-complement of a collection of simple parts. I could also allow Plastering to fill part of Fulcher's model. At the time of this writing I can reliably generate "meshes" for examples such as these, but I can not automatically get good-quality nodal positions. In many cases I have been able to get good quality meshes by positioning nodes by hand, so I do not think the connectivity is fundamentally bad.

5. Extensions and Non-template methods

There are several natural variations on the construction to explore. The first variation is that the surfaces could be completed in a different way. In particular, perhaps Whisker Weaving or another algorithm could complete the STC of the diced hex mesh in another way, so that the surfaces do not need to curve so sharply.

A second variation is that the duals could be pushed farther together. In particular, currently every quad face on the interface gets split into a geode-template. Quality is less than ideal where two interface quads have a small or large dihedral angle. This can occur even when the meshes are relatively structured; see Figure 9. It would be nice to "round off" these corners by pushing a dual surface into the opposite dual, so that the dual surface curves less. This method no longer produces pure templates. Figure 10 shows a two dimensional example of this.

Figure 9. Interface angles affect the quality of the geode-templates. Right, in three-dimensions it may be impossible to create a buffer layer: Suppose interface node *V* has six attached interface faces as in the figure. Since no point can see both of S_1 and S_2, one of the geode-templates attached to S1 or to S2 must have a non-positive jacobian at *V*.

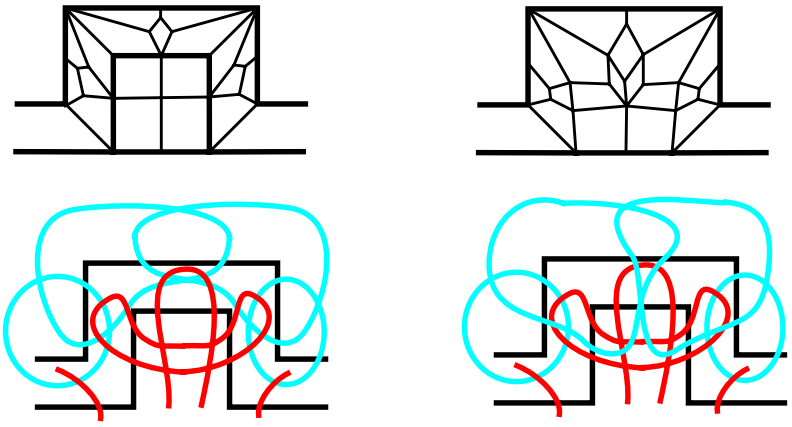

Figure 10. Pushing the arrangements so that they overlap further can improve quality, but modifies the pure templates. Left shows the pure templates. Right shows the result of puffing out two curves so that they are more round. Top shows the primal mesh, bottom shows the dual curves.

A third variation is to change the interface mesh: Split each interface quad into four triangles instead of two. The dual of the triangular mesh is still composed of circles and may be completed into spheres. I speculate that it is straightforward to carry out the geode-template construction steps. The biggest unknown is whether the resulting arrangement will dualize to a well-defined hex mesh. If not, then Mitchell[5] and Folwell[2] show how to fix-up the arrangement, but mesh quality will be poorer.

A fourth variation is to try to transition between two wildly different meshes, say two hex meshes with different element sizes that do not even conform node-wise. Completing the duals is the same as before. However, I would need to develop a new algorithm for pushing the duals together in a general way, and in many cases mesh fix-up as in Mitchell[5] and Folwell[2] would be required.

5.1 Why are Other Templates Hard?

In short, I designed the geode-template to be easy to mesh by keeping the dual curves of the boundary-mesh separate, so that no curve passes through a quad of both a diced-tri and a diced-quad. I also kept each dual curve *simple* (non-self-intersecting). In contrast, consider filling Schneiders's open problem,[9] Figure 11, with hexahedra. The surface mesh is identical to the variation of the geode-template with a pre-diced quad divided into four triangles, but without any sides. This problem, publicized several years ago, was difficult to fill at all, and no good-quality all-hex mesh is known. Figure 11 right shows the dual curves, one of which self-intersects eight times and passes through every quad of the boundary mesh. The octahedron has a boundary-mesh whose dual is just that curve.[1]

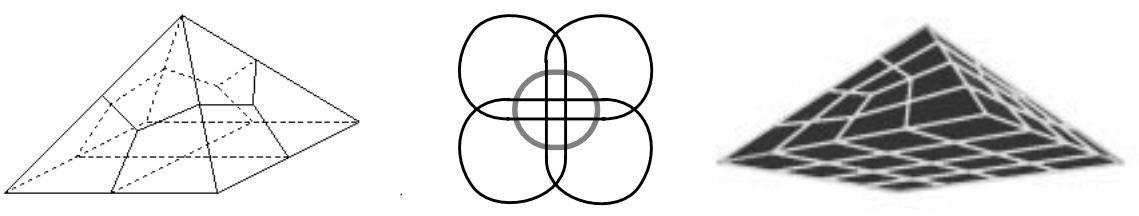

Figure 11. Schneiders's open problem.[9] No good all-hex mesh for the left figure is known. Center shows the two dual curves of the surface mesh laid flat. Right shows an underside-view of the problem after dicing the surface mesh; this version was meshed with the geode-algorithm.

On the CUBIT project, we have tried several pyramid-like templates. In these examples, unless sides like the geode-template's are introduced, the dual curves have a complicated self-intersection structure.

6. Conclusions

I have shown how to topologically conform a diced hexahedral mesh to a diced tetrahedral mesh by inserting an all-hex transition layer. The general method may be extensible to other transition problems. My results are practical, in that the algorithm is simple and the template can have good-quality hexes; at SNL, we are currently working on creating a geometrically-good transition layer for general boundaries.

Some interesting open problems remain. The geode-template indicates that, given the freedom to modify the surface mesh by dicing and the existence of a buffer layer, most surface meshes admit a well-shaped compatible hexahedral mesh. What if the surface mesh can not be diced or modified? Note that dicing gives a surface mesh whose node-edge graph is bipartite, which is sufficient to prove that a conforming hexahedral mesh exists, see Eppstein[1], but there is no guarantee that a good-quality mesh exists. Without dicing, the interface mesh may be composed of an odd number of quadrilaterals, which is impossible to fill with hexahedra.[5] Can the extent of dicing or modifying the surface mesh be limited in practical settings?

The buffer layer arises naturally as the dual of the intersection of the completed STC of the diced-tet-mesh with the completed STC of the diced-hex-mesh. Certain interface geometries do not admit a well-shaped buffer layer. One

interpretation of this is that the completed STC surfaces must curve too much. Could the STCs be pushed so that they overlap more, allowing more slowly curving surfaces?

Acknowledgments

I would like to thank Robert W. Leland and Timothy J. Tautges for suggesting this problem and developing its context. Thanks also go to Robert Schneiders and Christa Polaczek for discussing and publicizing Schneiders's open problem.[9]

References

[1] D. Eppstein, Linear Complexity Hexahedral Mesh Generation, *Proceedings, 12th Annual ACM Symposium on Computational Geometry*, Philadelphia, pp. 58-67, May 1996.

[2] N. Folwell and S. Mitchell, Reliable Whisker Weaving via Curve Contraction, submitted to *7th International Meshing Roundtable*.

[3] P. Knupp, Optimized-jacobian Based Smoothing, in progress. pknupp@sandia.gov.

[4] R. Meyers and P.Tuchinski, The "Hex-Tet" Hex Dominant Meshing Algorithm as Implemented in CUBIT, submitted to *7th International Meshing Roundtable*.

[5] S. Mitchell, A Characterization of the Quadrilateral Meshes of a Surface Which Admit a Compatible Hexahedral Mesh of the Enclosed Volume, *Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science*, Springer, pp. 465-476, 1996. Also http://endo.sandia.gov/~samitch/exist-abstract.html.

[6] P. Murdoch, The Spatial Twist Continuum: A Dual Representation of the All Hexahedral Finite Element Mesh, Ph.D. dissertation, Mech. Engr., Brigham Young University, December 1995.

[7] P. Murdoch and S. Benzley, The Spatial Twist Continuum, *Proceedings, 4th International Meshing Roundtable*, Sandia National Laboratories, pp. 243-251, October 1995.

[8] J. Robinson, CRE Method of Element Testing and the Jacobian Shape Parameters, Eng. Comput. Vol. 4, No. 2, p113-118, June 1987.

[9] R. Schneiders, An interesting open problem, http://www-users.informatik.rwth-aachen.de/~roberts/open.html.

[10] R. Leland, D. Melander, R. Meyers, S. Mitchell and T. Tautges, The Geode Algorithm: Combining Hex/Tet Plastering, Dicing and Transition Elements for Automatic, All-Hex Mesh Generation, submitted to *7th International Meshing Roundtable*.

[11] P. Tuchinsky, The Hex-Tet Hex Dominant Automesher: An Interim Progress Report, *Proceedings, 6th International Meshing Roundtable*, Park City, Utah, pp 183-13, October 1997.

Appendix A. Mesh Connectivity and Positions

This section describes how to reconstruct the geode-template. The following tables list the node positions and hex connectivity of a 24 x 14 x 24 geode-template centered at the origin.

Node	x	y	z	Node	x	y	z	Node	x	y	z
1	-12.0	7.0	12.0	17	12.0	7.0	0.0	33	3.9	-1.8	-3.6
2	12.0	7.0	-12.0	18	12.0	1.7	5.0	34	-3.9	-1.8	3.7
3	0.0	7.0	0.0	19	12.0	4.5	0.0	35	-4.1	-1.5	-4.1

Table 2: Node positions for a 24 x 14 x 24 geode-template centered at the origin.

Node	x	y	z	Node	x	y	z	Node	x	y	z
4	12.0	7.0	12.0	20	12.0	1.7	-5.0	36	-0.2	4.5	-5.3
5	0.0	7.0	12.0	21	12.0	-7.0	-12.0	37	-8.9	3.9	-6.5
6	12.0	7.0	0.0	22	0.0	-7.0	-12.0	38	-4.3	4.2	-2.3
7	4.6	7.0	4.6	23	5.0	1.7	-12.0	39	0.0	3.5	0.0
8	-12.0	7.0	-12.0	24	0.0	4.5	-12.0	40	4.4	4.3	2.3
9	-12.0	7.0	0.0	25	-5.0	1.7	-12.0	41	8.5	3.9	6.6
10	-12.0	-7.0	-12.0	26	0.0	7.0	-12.0	42	0.2	4.5	5.4
11	-12.0	-7.0	0.0	27	0.0	-7.0	12.0	43	4.1	-1.5	4.2
12	-12.0	1.7	-5.0	28	-5.0	1.7	12.0	44	5.7	0.7	0.7
13	-12.0	4.5	0.0	29	0.0	4.5	12.0	45	2.6	1.4	2.1
14	-12.0	1.7	5.0	30	5.0	1.7	12.0	46	0.0	0.0	0.0
15	-12.0	-7.0	12.0	31	0.0	-7.0	0.0	47	-2.6	1.4	-2.0
16	12.0	-7.0	12.0	32	-4.6	7.0	-4.6	48	-5.6	0.7	-0.6

Table 2: Node positions for a 24 x 14 x 24 geode-template centered at the origin.

Hex	n1	n2	n3	n4	n5	n6	n7	n8	Hex	n1	n2	n3	n4	n5	n6	n7	n8
1	17	20	2	21	31	33	23	22	14	48	13	12	35	34	14	11	31
2	43	18	17	31	44	19	20	33	15	28	27	15	1	34	31	11	14
3	43	30	27	31	18	4	16	17	16	41	5	1	14	42	29	28	34
4	36	24	26	37	33	23	2	20	17	38	32	9	13	39	3	1	14
5	37	26	32	38	20	2	3	39	18	40	7	3	39	41	5	1	14
6	19	6	2	20	40	7	3	39	19	46	39	14	34	45	40	41	42
7	45	40	19	44	46	39	20	33	20	47	38	13	48	46	39	14	34
8	46	39	20	33	47	38	37	36	21	36	24	25	35	37	26	8	12
9	33	23	22	31	36	24	25	35	22	47	38	37	36	48	13	12	35
10	48	47	46	34	35	36	33	31	23	37	26	8	12	38	32	9	13
11	34	46	45	42	31	33	44	43	24	41	5	4	18	40	7	6	19
12	31	22	10	11	35	25	8	12	25	42	41	18	43	45	40	19	44
13	42	29	28	34	43	30	27	31	26	42	29	30	43	41	5	4	18

Table 3: Connectivity of a geode-template: Nodes in hexes.